14

Haga lo que hicimos en aula: más sobre relacionamientos

En esta primera aula realizamos una modificación en el archivo persistence.xml donde configuramos las propiedades que nos dan acceso a la base de datos. En el curso anterior habíamos configurado la base de datos en la memoria, en esta parte vamos a almacenar los datos en un archivo local que nos va a permitir observar las modificaciones que estamos realizando.

```
<persistence-unit name="tienda" transaction-type="RESOURCE_LOC
    <properties>
        <property name="javax.persistence.jdbc.driver" value="
        <property name="javax.persistence.jdbc.url" value="jdb
        <property name="javax.persistence.jdbc.user" value="sa
        <property name="javax.persistence.jdbc.password" value

        <property name="hibernate.dialect" value="org.hibernat
        <property name="hibernate.show_sql" value="true"/>
        <property name="hibernate.format_sql" value="true"/>
        <property name="hibernate.hbm2ddl.auto" value="create-

    </properties>
```

COPIA EL CÓDIGO

- Vamos a crear dos nuevas entidades que nos van a permitir ampliar nuestro modelo inicial donde registramos un producto por categoría, ahora podremos registrar los pedidos que realizan diversos clientes de estos productos, por lo tanto vamos a crear la entidad Pedido y la entidad cliente que estarán relacionados con la anotación `ManyToOne` donde un cliente tendrá muchos pedidos similar a la entidad Producto con Cliente

```
@Entity
@Table(name="pedidos")
public class Pedido {
        @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Long id;
    private LocalDate fecha=LocalDate.now();
    @Column(name="valor_total")
    private BigDecimal valorTotal;

    @ManyToOne
    private Cliente cliente;

@OneToMany(mappedBy="pedido", cascade=CascadeType.ALL)
private List<ItemsPedido> items=new ArrayList<>();

public void agregarItems(ItemsPedido item) {
        item.setPedido(this);
        this.items.add(item);
    }

    public Pedido(Cliente cliente) {
        this.cliente = cliente;
    }
    public Pedido() {}
    //getters / setters
}
@Entity
@Table(name="clientes")
public class Cliente {

    @Id
```

```
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Long id;

    private String nombre;
    private String dni;

    public Cliente() {}

    public Cliente(String nombre, String dni) {
        this.nombre=nombre;
this.dni=dni;
    }

    //getters / setters
}
```

COPIA EL CÓDIGO

- Para conseguir relacionar los pedidos de los clientes a los productos podemos crear un atributo con la entidad `@ManyToMany` y jpa crea automaticamente una nueva entidad, en nuestro caso tenemos que crear una nueva entidad y realizar un relacionamento bidireccional para que esa entidad intermedia contenga atributos auxiliares.

```
@Entity
@Table(name="items_pedido")
public class ItemsPedido {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Long id;

    private int cantidad;
```

```
    private BigDecimal precioUnitario;

    @ManyToOne
    private Producto producto;

    @ManyToOne
    private Pedido pedido;

    public ItemsPedido() {}

    public ItemsPedido(int cantidad, Producto producto, Pedido
        this.cantidad = cantidad;
        this.producto = producto;
        this.pedido = pedido;
        this.precioUnitario=producto.getPrecio();
    }

}
```

COPIA EL CÓDIGO

- Como vamos a guardar nuevos registros debemos crear clases DAO para estas nuevas entidades excepto para ItemPedido donde JPA reconoce que es un relacionamiento bidireccional que conecta un grupo de entidades con otro.

```
public class PedidoDao {

    private EntityManager em;

    public PedidoDao(EntityManager em) {
        this.em = em;
    }
```

```
    public void guardar(Pedido pedido) {
        this.em.persist(pedido);
    }

…
```

COPIA EL CÓDIGO

```
public class ClienteDao {
    private EntityManager em;

    public ClienteDao(EntityManager em) {
        this.em = em;
    }

    public void guardar(Cliente cliente) {
        this.em.persist(cliente);
    }
…
```

COPIA EL CÓDIGO

```
public class RegistroDePedido {
    public static void main(String[] args) throws FileNotFound
        registrarProducto();

        EntityManager em = JPAUtils.getEntityManager();
        ProductoDao productoDao = new ProductoDao(em);
        Producto producto = productoDao.consultaPorId(1l);
        ClienteDao clienteDao = new ClienteDao(em);
        PedidoDao pedidoDao = new PedidoDao(em);
```

```
        Cliente cliente = new Cliente("Juan","k6757kjb");
        Pedido pedido = new Pedido(cliente);
        pedido.agregarItems(new ItemsPedido(5,producto,pedido)

        em.getTransaction().begin();
        clienteDao.guardar(cliente);
        pedidoDao.guardar(pedido);

        em.getTransaction().commit();

    }
…}
```

COPIA EL CÓDIGO